Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

A BRIEF INTRODUCTION TO COQ

The Coq Proof Assistant and Schedulability Analysis

March 13, 2020

Tanja Almeroth

Studienbereich DCSM
Hochschule **RheinMain**

TABLE OF CONTENTS

RECAP

LAST TIME

- → proof assistants
- → functional programming in Coq
- → applications of Coq
- → PROSA and RT-proofs

FORMER OUTLOOK

- → lecture notes for Steffen Reith, incorporate review
- → incorporate linear temporal logic (LTL)
- → Steffen Reith's review
- → incorporate Steffen Reith's review on the PROSA working directory
- → tutorial on `Gallina` and `SSreflect` (4 person days)
- → formally prove Kaiser's EDF scheduler in PROSA hopefully less then ($\approx 1'320$ LOC)
- → look at the best approach to validate a complete OS kernel

FORMALLY PROVE KAISER'S EDF-SCHEDULER

PROSA: FORMALLY PROVEN SCHEDULABILITY ANALYSIS

→ basic tools from logic
→ precise claims about programs
→ functional programming methods of programming and logical reasoning about programs

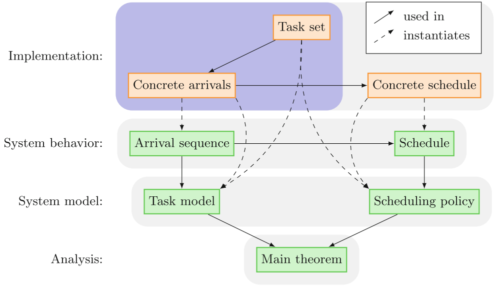

Figure: PROSA layers. Source: [1]

MAIN THEOREM OF THE PROSA ECRTS16 ARTIFACT EVALUATION

practical specification: correctness of the proof

→ EDF-specific interference bound definition and proof of termination and correctness of Bertongna and Cicerei's response time analysis RTA for FP scheduling

→ **definition and proof of termination and correctness of Bertongna and Cicerei's RTA for EDF scheduling ($\approx 1'320$ LOC)**

my current working directory:
https://gitlab.cs.hs-rm.de/almeroth/prosa_working_dir
due to documentation including all listings
($\approx 13'070$ LOC)

LINEAR TEMPORAL LOGIC (LTL) AND MULTIPROCESSOR SCHEDULING

LTL AND MULTIPROCESSOR SCHEDULING

taxonomy of multiprocessor scheduling by David and Burns [2]

- → *fixed task priority*
  Each task has a fixed priority applied to all of its jobs.
- → *fixed job priority*
  The jobs of a task may have different priorities. But each job's priority is single and static. e.g. EDF scheduling.
- → *dynamic priority*
  A single job has different priorities at different times. e.g. least laxity first (LLF) scheduling.

Idea: Apply LTL to verify static and dynamic priority scheduling methods.
Conjecture: Predicate logic-based approaches fail for dynamic priority scheduling methods.

WHAT IS LTL?

- → Linear temporal logic is linear like a linear tree (graph)
- → apply this linearity approach to a discrete time in a state-space system
- → LTL's origins are in the natural language descriptions
- → natural numbers $\mathbb{N}$ represent a moment in time

Example: typical temporal operators (source: see [1])

- → $\bigcirc\phi$ means »ϕ is true in the next moment«
- → $\Box\phi$ means »ϕ is true in all future moments«
- → $\Diamond\phi$ means »ϕ is true in some feature moments«
- → $\phi\mathcal{U}\psi$ means »ϕ is true until ψ is true«

Goal: Verification of dynamic priority scheduling methods.
Note: Functional verification means verification during run-time.

MULTIPROCESSOR SCHEDULING

PROBLEM

In ***hard real-time systems*** we say a process or task is defined as an sequential execution of a program or a job on a processor. The execution ends after a ***finite number of steps***.
»One big ***train*** of commands has to pass in a finite amount of time.«

Definition

Multiprocessor scheduling can be viewed as attempting to solve two problems [2].

(i) The ***allocation problem***, or on which processor the task should execute.

(ii) The ***priority problem***, or in when, and in what order with respect to jobs of tasks, each job should execute.

PRIORITY PROBLEM

Kasier's encapsulated EDF-scheduler [3]

- → real-time as in [1]
- → hierarchical (local and global distinction)
- → multi-core (in the real-time-sense)
- → earliest deadline first (EDF) on a local level
- → fixed priority (FP) on a global level
- → sporadic
- → disruptive

Main theorem: correctness of analysis although it is a deductive derivation

PROBLEM: DEFINE A JOB OR TASK

Definition

Let

$\delta_p \in \mathbb{N}$ be a periodic disruptive process and
$\delta_s \in \mathbb{N}$ be an asporadic disruptive process.
Let $P := \{1, \cdots, n\} \subset \mathbb{N}^n$
be a job, a set of disruptable processes
with fixed priority order ascading.
Let $\delta p_i \in \mathbb{N},$ for $i = 1, .., n$ denote the period and
$\delta e_i \in \mathbb{N}$ for $i = 1, .., n$ denote the execution time of a proces.

MULTIPROCESSOR SCHEDULING ALGORITHM

Definition

We say a multiprocessor scheduling algorithm σ is given by

$$\sigma : \mathbb{N}^n \times \mathbb{N}^n \longrightarrow \mathbb{N} \times \{1, 2, \cdots, m\} \quad (1)$$

$$\forall i, \delta e_j \in \mathbb{N} \quad (i, \delta e_j) \mapsto (t, k) \in \mathbb{N} \times \{1, 2, \cdots, m\}. \quad (2)$$

map time i and execution time δe_i to time-sequence t and processor k

MULTIPROCESSOR SCHEDULING ALGORITHM

Definition

A scheduler satisfying the EDF-regulation

$$U(P) = \sum_{i=1}^{n} \frac{\delta e_i}{\delta p_i} \leq \frac{\Delta e_{sv}}{\Delta p_{sv}} \tag{3}$$

is called feasible.

Theorem

A hard-real-time, hierarchical, multi-core, EDF, sporadic and disruptive time scheduler is feasible iff

$$\sum_{i=1}^{n} \lfloor \frac{t}{\Delta p_i} \rfloor \Delta e_i \leq \lfloor \frac{t}{\Delta p_{sv}} \rfloor \Delta e_{sv} + \max(0, t mod \Delta p_{sv} - \Delta e_s - \Delta e_p). \tag{4}$$

OUTLOOK

OUTLOOK

- lecture notes for Steffen Reith, incorporate review
- incorporate Steffen Reith's review on the PROSA working directory (if the review is given)
- formally proof the Kaiser's EDF scheduler in PROSA hopefully less then ($\approx 1'320$ LOC)
- apply the theorem of Akra -Bazzi
- tutorial on `Gallina` and `SSreflect` (4 person days)

BIBLIOGRAPHY

- B. C. Pierce and A. A. de Amorim and C. Casinghino and M. Gaboardi and M. Greenberg and C. Hrițcu and V. Sjöberg and B. Yorgey
  »Software Foundations, Logical foundations, Volume 1«
  https://softwarefoundations.cis.upenn.edu/current/lf-current/index.html, 2019

- Coq- Project Website
  »The Coq Proof Assistant«
  https://coq.inria.fr , 2019-01-09

- RT-Proofs Website
  »Formal Proofs for Real-Time Systems«
  https://rt-proofs.inria.fr, 2020-14-02

BIBLIOGRAPHY

- Coq Integrated Development Environment - Official Documentation
  »Coq Integrated Development Environment«
  https://coq.inria.fr/refman/practical-tools/coqide.html
- Proof General - Project Website
  »Proof General, a generic interface for proof assitants.«
  https://proofgeneral.github.io/, 2020-14-02
- Coquille - Andreas Werner's Fork
  »Coquille, a vim plugin.«
  https://github.com/Werner2005/coquille, 2020-14-02

BIBLIOGRAPHY: ONLINE

N. Giannarakis
Coq - Syntax Highlighting
»Personal GitHub Profile«
https://github.com/nickgian/thesis/lstcoq.sty, 2019-19-09

BIBLIOGRAPHY: REAL-TIME SYSTEMS

- R. Kaiser and K. Beckmann and R. Kröger
  »Echtzeitplanung«
  Handouts **https://www.cs.hs-rm.de/~kaiser/1919_ezv/6_Scheduling-handout.pdf** 2020-07-01
- J. W.S. Liu
  »Real-time Systems«
  Prentice-Hall, Inc., ISBN: 0-13-099651-3, 2000
- R. Kasier
  »Virtualisierung von Mehrprozessorsystemen mit Echtzeitanwendungen«
  PHD Thesis, Universität Koblenz-Landau, 11-02-2009

BIBLIOGRAPHY: REAL-TIME SYSTEMS

G. Bollella
»Slotted Priorities: Supporting Real-Time Computing Within General-Purpose Operating Systems«
PHD Thesis, Chapel Hill, 1997

BIBLIOGRAPHY: REAL-TIME SYSTEMS

F. Cerqueira and F. Stutz, and B. Brandenburg
»Prosa - ECRTS'16 Artifact Evaluation«
https://prosa.mpi-sws.org/releases/v0.1/artifact/,
2020-01-09

F. Cerqueira and F. Stutz, and B. Brandenburg
»PROSA: A Case for Readable Mechanized Schedulability Analysis«
https://www.mpi-sws.org/~bbb/papers/pdf/ecrts16f.pdf
Proceedings of the 28th Euromicro Conference on Real-Time Systems (ECRTS), 2016

BIBLIOGRAPHY: COMPUTER AND COMMUNICATION SECURITY

Almeida, José Bacelar and Barbosa, Manuel and Barthe, Gilles and Blot, Arthur and Grégoire, Benjamin and Laporte, Vincent and Oliveira, Tiago and Pacheco, Hugo and Schmidt, Benedikt and Strub, Pierre-Yves
»Jasmin: High-Assurance and High-Speed Cryptography«
Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS), 2017

BIBLIOGRAPHY: VERIFYING AN OS KERNEL

- Guo, Xiaojie and Lesourd, Maxime and Liu, Mengqi and Rieg, Lionel and Shao, Zhong
  »Integrating Formal Schedulability Analysis into a Verified OS Kernel«
  Computer Aided Verification, Springer International Publishing, 2019
- Davis, Robert I. and Burns, Alan
  »A Survey of Hard Real-Time Scheduling for Multiprocessor Systems«
  Association for Computing Machinery, 2011

Alessandro Artale
»Formal Methods Lecture III: Linear Temporal Logic«
lecture slides, found online at
www.inf.unibiz.it/~artale/FM/fm.htm
Faculty of Computer Science - Free University of Bolzano